**Test Report on Real Time Event Management System**

**Name: Ravindran Dharshan**

**IIT NO : 20220693**

**UoW NO : w1953513**

**Overview**

The following report documents the test scenarios for the TicketPool class. Each test case ensures that the functionality adheres to defined requirements. The results are organized with detailed observations, expected outcomes, and any resolutions for identified issues. Additionally, summary tables and a deeper analysis are included.

**Test Case Details**

**Test 1: Adding Tickets to the Pool**

* **Scenario:** Add tickets to the pool and confirm the count matches the number of tickets added.
* **Input:**
  + Vendor: Vendor1
  + Tickets Added: 50
* **Expected Outcome:** The ticket pool size should increase to 50 tickets.
* **Actual Outcome:** Ticket pool size reached 50 tickets.
* **Result:** ✔ Passed
* **Comments:** Functionality performed as expected with no issues.

**Test 2: Exceeding Max Ticket Capacity**

* **Scenario:** Ensure that the ticket pool does not exceed the defined maximum capacity.
* **Input:**
  + Vendor: Vendor1
  + Tickets Added: 100 (exceeds capacity)
* **Expected Outcome:** Ticket pool size remains at the maximum capacity of 50.
* **Actual Outcome:** Ticket pool size correctly capped at 50.
* **Result:** ✔ Passed
* **Comments:** The system properly limited the pool size to the maximum defined capacity.

**Test 3: Purchasing a Ticket**

* **Scenario:** Verify that a single ticket purchase reduces the pool by one and increments the tickets sold counter.
* **Input:** None (simulates a single ticket purchase).
* **Expected Outcome:**
  + Ticket pool decreases by 1.
  + Tickets sold counter increases by 1.
* **Actual Outcome:**
  + Ticket pool size decreased to 49.
  + Tickets sold counter incremented to 1.
* **Result:** ✔ Passed
* **Comments:** Proper synchronization of pool size and tickets sold was observed.

**Test 4: Multiple Customers Purchasing Tickets**

* **Scenario:** Simulate 10 customers purchasing tickets and validate that ticket pool and sales are updated correctly.
* **Input:**
  + Initial Ticket Pool Size: 50
  + Customers Purchasing: 10
* **Expected Outcome:**
  + Tickets sold counter increases by 10.
  + Ticket pool decreases by 10.
* **Actual Outcome:**
  + Tickets sold counter increased to 10.
  + Ticket pool decreased to 40.
* **Result:** ✔ Passed
* **Comments:** Logic handled multiple ticket purchases accurately. Iterative improvements ensured edge case compatibility.

**Test 5: Interrupt Simulation**

* **Scenario:** Confirm that adding tickets is halted when the simulation is interrupted.
* **Input:**
  + Tickets Added: 10
* **Expected Outcome:** No new tickets are added after the interrupt.
* **Actual Outcome:** Ticket pool size remained unchanged after the interrupt.
* **Result:** ✔ Passed
* **Comments:** Simulation halt functionality behaved as expected.

**Analysis and Observations**

1. **Strengths:**
   * The ticket pool functionality demonstrated robustness in handling basic operations like adding and purchasing tickets.
   * System accurately enforced the maximum capacity limits.
   * Interrupt simulation worked flawlessly, preventing unintended changes.
2. **Areas of Improvement:**
   * Additional edge cases like zero tickets available or simultaneous customer requests could be tested to enhance reliability.
   * Error handling during purchase (e.g., pool already empty) could be validated.
3. **Issue Resolution:**
   * **Issue in Test 4:** Initially, customers attempting to purchase tickets when fewer tickets remained were not handled well.

* **Resolution:** Logic was updated to correctly limit purchases to available tickets.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Expected Outcome | Actual Outcome | Result | Remarks |
| Adding Tickets to the Pool | Pool size equals tickets added. | Pool size reached 50. | ✔ Passed | Functionality worked as expected. |
| Exceeding Max Capacity | Pool size remains at max capacity (50). | Pool size capped at 50. | ✔ Passed | Pool size limited appropriately. |
| Purchasing a Ticket | Pool size -1, tickets sold +1. | Pool size 49, tickets sold 1. | ✔ Passed | Sales and pool updates synchronized. |
| Multiple Customers Purchasing | Pool size -10, tickets sold +10. | Pool size 40, tickets sold 10. | ✔ Passed | Multiple purchases handled correctly. |
| Interrupt Simulation | Adding tickets blocked after interrupt. | Pool size unchanged after interrupt. | ✔ Passed | Simulation stop functionality worked well. |

This detailed testing report confirms that all core functionalities of the TicketPool class meet the specified requirements. The system is now ready for deployment with strong foundational tests in place.